```
__________________________ //  ________________________________
                          /##/
  //////                  ##/  /////      ////////    //////
//######//   //          //  //#####//  //#########/ //#######//
##/     ##/  ##/         ##/ ##/    ##/ ##/         ##/     ##/
##/     ##/  ##/         ##/ ##/    ##/ ##/         ##/     ##/
##/     ##/  ##/         ##/ ##/////##/ ##//////    ##//////##/
##/ //  ##/  ##/         ##/ #######/   ########/   ########/
##/ ##/ ##/  ##/         ##/ ##/        ##/         ##/##/
##/  ##/##/  ##/         ##/ ##/        ##/         ##/ ##/
##////##/    ##////////  ##/ ##/        ##////////  ##/  ##/
  ####/##/     #########/ ##/ ##/          #########/ ##/   ##/
        ##/
________ ##/ __________________________________________________
```

Número 37 - Marzo/Abril de 1992 - Año 4 - Cuarta época

CONTENIDO:

Revista bimestral en disco del grupo español de usuarios de QL QLíper.
Dumonata diskbulteno de la hispana QL-uzantaro QLíper.
Bimestral disk magazine of the Spanish QL users group QLíper.

QLíper acepta artículos en español, esperanto o inglés.
QLíper akceptas artikolojn en la hispana, esperanta aux angla.
QLíper accepts articles in spanish, esperanto or english.

Si quieres más información, contacta con el redactor:
Se vi volas plian informon, kontaktu la redaktiston:
If you want more information, contact the editor:

QLíper
Marcos CRUZ
Acacias 44 (Monteclaro)
E-28023 MADRID
(España/Hispanujo/Spain)
Número de teléfono/Telefonnumero/Telephone number: +34-(9)1-7153873

NOTAS DE LA REDACCION

El envío de este número se ha retrasado tres días; no es mucho, ¿no? Además, en vez de quejaros, poneos a rellenar con interesantes octetos algunos aburridos sectores de algún polvoriento disco y mandádmelo, porque el día en que (San) Salvador coja la fiebre amarilla no sé qué va a ser de esto. Queda algún material para el 38, no obstante, como siempre.

Réquiem por Microhobby. Se veía venir, pero aun así ha sido un golpe bajo a nuestros sentimientos... Mi colección totalmente completa llora desconsolada lágrimas de nostalgia. Aún queda la esperanza de verla aparecer de nuevo, quizá bimestral, quizá con menos páginas, pero nada hay seguro. Como suscriptor, recibí un cheque de 436 pesetas por el número que me faltaba, y espero recibir una nueva carta con la decisión final sobre la superviviente pionera de las revistas de microordenadores.

"International QL Report", de Estados Unidos, me ha mandado una lista de términos informáticos en inglés con la petición de que les envíe la traducción al castellano. Según explican, el número de programas de dominio público para QL ha aumentando impresionantemente en los últimos dos años (mencionan las librerías inglesa, francesa, alemana, española, danesa e italiana), y cada programador usa en ellos (¡faltaría más!) su propia lengua materna, con lo cual el caos es ya alucinante, y la mayoría de la gente no puede usar muchos programas debido al problema lingüístico. Al parecer a estos señores no se les ha ocurrido otra idea mejor ni más eficaz que hacer un rudimentario diccionario en 6 idiomas, para traducir, entre un "sálvese quien pueda" y un "tonto el último", los mensajes de pantalla y las instrucciones... Pero como por un lado su conocimiento de otros idiomas es ridículamente rudimentario, como cabe esperar, y por otra parte los que se supone que sí conocen más o menos bien algún otro idioma, según me confiesan en su carta, no tienen ni idea de ordenadores, pues el problema es insoluble para ellos.

En fin, los esperantistas nos partimos de risa con estas cosas. Pero después nos dan ganas de llorar... No creo que ninguno de vosotros tenga amigos en los cinco continentes, ni haya conversado multitud de veces con toda naturalidad con gente de un montón de países a la vez, sin ningún tipo de problema y después de sólo 6 meses de estudio. Si hubieseis experimentado eso, sabríais por qué nos partimos de risa y a la vez lloramos cuando vemos que siguen empeñados, por simple y pura ignorancia, en matar dragones con matamoscas (desde el punto de vista de la eficacia), y moscas a cañonazos (desde el punto de vista de los costes).

Si la experiencia es la madre de la ciencia, la ignorancia es la madre de la intolerancia (léase prejuicios): El mundo tardó 200 años en aceptar el sistema métrico decimal como norma estándar, y aún hoy los anglosajones se resisten. Hace cien años había muchos reputados y famosos "sabios" que decían que usar metros y centímetros era ilógico y absurdo, que era más natural usar pies, pulgadas, varas, codos, palmos...

Hoy a todos nos producen risa esos argumentos, y no podemos entender cómo la gente se resistía a aceptar algo tan evidentemente útil y eficaz, que produce tantas ventajas a la sociedad, ¿no es así? Pues es el mismo caso, amigos. La única diferencia es que los esperantistas ya vamos muy por delante en el terreno de la comunicación, y por eso nos reímos por anticipado.

Y para colmo, por si alguno no lo sabe, diccionarios multilingües sobre informática y electrónica hay varios, editados en Hungría y Japón, y una de sus lenguas es el esperanto. Pero al parecer hay quien quiere reinventar la rueda, y después vendérnosla.

En fin, perdonad el inciso, y volvamos a lo nuestro:

Encontraréis en el fichero T87INST_T87 la traducción al castellano, con sus muchos anglicismos incluidos, del text87 3.00, que me envió Salvador. Por lo demás, no encontraréis el texto de la carta que recibimos desde Chile, y de la que os hablé en el número 37, porque se me ha traspapelado en algún agujero negro doméstico; tal vez más adelante.

Y como postre de este número podréis degustar algo que quedó pendiente en el número anterior, el artículo sobre la batalla por suceder al código ASCII; que lo disfrutéis, porque no es para menos.

Sólo una petición práctica a los (pocos) colaboradores: Los textos, cuanto

más claros y limpios mejor, sin mezclas de tipos de letras, sin cambios de márgenes, etc. Y los ficheros de los discos lo más ordenados posible, porque a veces es un lío saber qué fichero pertenece a qué programa. Gracias.

Ah, los movimientos de la caja de QLíper aparecerán a partir de ahora en un fichero aparte, cuentas_doc, así como la lista de suscriptores, que pasa a aparecer en formato del Quill en vez de en ASCII, con el nombre suscript_doc. Están grabados justo después del fichero de portada de la revista, pero no hay que unirlos a ésta.

Hasta la próxima, en mayo.

Marcos, 1992 03 02

QUE SE SIENTE CUANDO SE RECIBE LA REVISTA

¡Aleluya, Aleluya,...! Por primera vez en tres años y 3 meses, he recibido nuestra revista, QLíper. Antes lo normal era prepararla y enviarla yo mismo. Al conocer el contenido completo de la revista por antemano, se perdía la ilusión de tener en nuestras manos algo nuevo.

También es una buena noticia que el club se reúna una vez cada dos meses en Madrid. Este era uno de los motivos por el cual decidí que el club debía trasladarse a Madrid, pues era el único lugar donde vivían suficientes socios como para que pudiesen hacer una reunión.

En la revista Sinclair QL World de enero me sorprendió ver el programa Catch Jack, la versión inglesa del programa Retrato Robot de José Carlos DE PRADA (publicado en español y en versión QLIBERATOR en CUQ 1). Ya era hora de que alguno de nosotros se decidiese a enviar colaboraciones a QL World. A diferencia de QLíper (ya sabéis que nuestro presupuesto es bajísimo), allí se pagan las colaboraciones muy bien.

Acabo de recibir el catálogo Apple de CMV Sistemas Informáticos, S.L. He comprobado que los MACs han bajado mucho de precio, y son una buena alternativa a los PCs. Sin embargo, el software y hardware para la gama MAC tiene unos precios que no sé quién de nosotros se los puede permitir.

Actualmente la gente compra un PC por razones de precio y gorronería. Al estar los precios de los PCs AT tan bajos, quizás lo que más le interese a los usuarios QL como nueva ampliación sea una tarjeta QL para PC basada en el MC 68040 y BUS VME. Esto supondría un equipo muy interesante.

En lo referente al club, os diré que me temía que mucha gente no iba a renovar, pero continuar en 1992 con socios nuevos y el mismo número de socios que en 1991, es una gran noticia. Espero que gracias a la sección Club Access de la revista Sinclair QL World nuestro número aumente un poquitín.

También os recuerdo que el número de socios no es un factor importante (recordad los cerca de 300 socios en la época de Qlave). Si los socios no colaboran, mucho me temo que cada vez sera más dificil llenar el disco. Colaborar no es tan difícil, socios como yo necesitamos ideas nuevas para escribir programitas, así como alterar o ampliar antiguos programas.

Salvador MERINO, Fuengirola, 1992 01 21

RECIBI LA NUEVA ROM 2.27 PARA LA GOLD CARD

Hoy he recibido la nueva ROM 2.27 para mi GOLD CARD. No solamente corrige los BUGs que tenía la v2.15 con las disqueteras ED, sino que provee unos nuevos comandos muy interesantes, los cuales son:

- DEV_USE número_dev, Directorio_real .- Crea un nuevo dispositivo llamado dev(1-8)_, el cual controla el directorio o dispositivo que nosotros deseemos.
- DEV_LIST.- Deja una lista de los dispositivos DEV configurados.
- SLUG.- Este comando es muy interesante para poder controlar algunos programas que son tan rápidos que no hay quien pueda manejarlos. Este comando lo que hace es reducir el tiempo de lectura del teclado. SLUG 15 es el más recomendable.

Salvador MERINO, Fuengirola, 1992 02 12

DISCOS ED

En España es muy difícil comprar discos ED, y si se encuentran, seguramente nos pedirán una fortuna (solamente hay que ver el precio de los discos HD).

Miracle me ha confirmado que vende discos ED a España en cajas de 10

unidades por 30 (gastos de envío incluidos).

Si comparamos el precio de los discos ED con los DD, nos daremos cuenta que la diferencia de precio no es muy apreciable. Cada disco ED equivale a 4.5 discos DD. Esto significa que al cambio actual de la libra, cada 720 Kbytes ED salen a unas 120 ptas. (El precio por Kbyte de los discos ED con marca es más barato que los discos DD con marca). En otras palabras, podemos utilizar discos ED sin arruinarmos, ya sea para beneficiarnos de su velocidad y capacidad como para usarlos como medio de almacenamiento de seguridad en archivos tipo ARC, o similar, para ahorrar espacio físico.

Salvador MERINO, Fuengirola, 1992 02 12

EL RINCON DEL PINCHADISCOS

A pesar de que poseemos una super-colección de programas en PD (algunos son viejos programas comerciales cedidos por sus autores), seguimos sin hacer uso de ella. Este año solamente he recibido un sólo pedido de Félix Alonso de 9 discos. El precio de copia por disco sigue siendo 100 ptas (el equivalente a los 50p) e incluye copia, embalaje y gastos de envío (el destinatario debe proporcionar el disco, o 225 ptas por disco -todo incluido-).

Las causas por la que no hacemos uso de la librería son:

- El usuario actual de QL posee demasiados programas.
- No sabemos exactamente qué hacen los programas. Se da el caso de que alguien compra un programa comercial, y existe uno en PD que hace lo mismo.
- El principal es que a todos nos cuesta trabajo ir a correos para enviar el paquete.

A pesar de todo, puedo garantizar que la librería QLíper seguirá creciendo (no los he contado, pero debe de haber más de 100 discos o sobre 150 contando los CUQs. O lo que es lo mismo, alrededor de 1.500 programas o 100 Mbytes de material), y no me importaría en un futuro, si el grupo creciese, ceder el puesto a nuevas librerías. En fin, tengo que reconocer que solamente somos 20 gatos con muchas ganas de rollo (¡Son ya 6 años!).

Salvador MERINO, Fuengirola, 1992 02 13

[En este QLíper encontraréis el fichero ANEXO_1_LIBRERIA_QLIPER_TXT, que contiene lo que su propio nombre indica. Nota de la redacción.]

LO ULTIMO SOBRE EL SUPER-QL

QLAW ya ha recibido 1.000 cuestionarios, y tiene una idea de lo que realmente deseamos los usuarios de QL como futura ampliación.

El prototipo de SuperQL actual utiliza en casi su totalidad periféricos de PC (excepto la nueva QL PCB), y acepta una tarjeta madre de PC (386 o superior). El sistema operativo elegido es la última versión del QDOS avanzado de Tony TEBBY (parte de él ya corre en la Gold Card).

Una caja con fuente alimentación, teclado, ratón, tarjeta VGA, controladores de disco flexible y de disco duro, 40 Mbytes de disco duro, una unidad de discos, un monitor monocromo VGA, y la tarjeta PCB 68000 a 16 MHz, podría costar alrededor de 800 (incluido VAT/IVA). Si a esto se le agrega una tarjeta madre de PC (386 o más) habría que sumar otras 300.

Hay planes para comprar a Tony TEBBY el dispositivo QFLP para utilizarlo con Minerva (sistema operativo fuera de los derechos de Amstrad) en el Commodore Amiga. El nuevo sistema compatible con QDOS para Amiga sería totalmente legal y se comercializaría en forma de ROM por unas cuantas decenas de libras.

Este año va a salir mucho hardware nuevo para nuestro viejo QL. Por lo pronto veremos con casi con toda seguridad: un mínimo de dos tarjetas gráficas, una caja con fuente alimentación y cable plano en la cual podremos instalar 3 tarjetas QL, y un nuevo interface de disco duro más rápido que el de Miracle.

Miracle tiene pensado construir una tarjeta QL para PC. Esto a mí me vendría de perlas, pues aunque estoy muy contento con mi Gold Card, me gustaría poder utilizar todos los periféricos baratos que hay en el mundo PC sin tener que sacrificar ni abandonar mi querido QDOS. La razón es sencilla. Aunque algún día sea una realidad el SuperQL, difícil veo encontrarlo disponible en la tienda de la esquina sin tener que esperar varios años a que se produzca tal milagro. Me es más fácil pedir una tarjeta QL para PC por correo, y comprar el PC cerca de mi casa. Además, hay que tener en cuenta que puedo comprar en Fuengirola un PC 486 a 33 MHz, 2 Mb RAM, disco duro 82 Mb, gráficos SVGA, monitor, teclado, ratón, MS-DOS 5.0, Microsoft WINDOW, varios programas de gestión... por menos de 300.000 ptas. Tenemos que descubrirnos: contra los precios tan bajos de los PCs no hay quien pueda. Eso sí, es una buena idea utilizar los periféricos estándar de PC para abaratar costes.

Salvador MERINO, Fuengirola, 1992 02 18

EL FINAL DE MICROHOBBY

Me ha sorprendido bastante ver en el Teletexto que la revista Microhobby ha desaparecido del mercado en su número 217.

Esta revista nació en noviembre de 1984. Entre los primeros suscriptores hizo un sorteo de varios QLs, Interfaces I, II, y Microdrives. Su salida fue semanal durante los años de plena euforia "sinclairiana". Sobrevivió a los años sucesivos de la caída de Sinclair pasando a quincenal, y más tarde, ya en plena decadencia, mensual.

Confieso que poseo toda la colección de Microhobby desde el primer número hasta finales de 1986, pues fue cuando la revista dejó abandonado el QL como si nunca hubiese existido. Eso sí, las noticias sobre el QL en la sección Panorama eran muy interesantes.

Todo esto me hace recordar revistas pioneras como ZX y Todospectrum con las cuales aprendí a escribir mis primeros programitas. Lástima que no tuvieran tanta suerte como Microhobby.

En la nota se dice que es probable que la revista vuelva a salir al mercado si los usuarios colaboran escribiendo material para llenarla, pues todo hace pensar que el problema de hoy es que ya casi nadie programa en España, y quien lo hace, es pensando en el lucro.

Yo empecé en 1979 como usuario de consolas de juegos Atari. Con los ZX80 y ZX81 podríamos decir que empezo el "BUM" de los microordenadores domésticos, y quien compraba uno de esos cacharros con un simbólico número de Kbytes, lo hacía para experimentar en el campo de la programación. Luego con la llegada del ZX Spectrum y toda su competencia, los microordenadores ganaron la batalla a las consolas de juegos, pues sus juegos eran más potentes y baratos, se podían programar, y utilizar en aplicaciones de negocios y otras. Si a todo esto agregamos que el precio de una consola de juegos y el de un microordenador era casi idéntico, esto supuso el fin de las consolas de juegos. Pero hoy en día los tiempos han cambiado muchísimo. La mayoría de los ordenadores de 8 bits han desaparecido del mercado. Ha nacido una nueva generación de consolas de juegos, pero me gustaría recalcar las portátiles, pues quizás sean las más atrayentes (principal "BUM" de la nueva generación de consolas). Los actuales ordenadores poseen juegos muchísimo más potentes que los que se puedan encontrar en cualquier consola, y el disco sigue siendo más barato que el cartucho ROM. Entonces habría que preguntarse el porqué de comprar una consola. La respuesta es sencilla, la mayoría de los PCs que se venden para usar en casa terminan usándose como ordenador de juegos, y los padres normalmente van buscando algo fácil de usar para sus pequeños retoños.

En resumen, Si Microhobby desaparece del mercado para siempre, será una gran pérdida y el fin de una era. Y el comienzo de una nueva era... Hobby Consolas.

Salvador MERINO, Fuengirola, 1992 02 18

COMO MONTAR LA GOLD CARD EN SPEM SYSTEM2

Como ya os dije en mi primer comentario, la Gold Card solamente funcionaba en la caja de Spem si se instalaba en el conector de expansión externo. Como es natural, hay que estar un poco pirado o majara para instalar la Gold Card ahí.

Escribi a Miracle Systems, pero no recibí respuesta.

Hace tiempo leí en Quanta que Miracle Systems le dijo a un usuario de caja Schön que debía quitar el viejo MC68008.

Hoy he recibido el catalogo de Spem de Otoño'91. Se dice que para montar la Gold Card en QL-System2 hay que hacer lo siguiente: Si puo' montare all'interno di un System2 Spem, si deve solo togliere il vecchio 68008.

No tengo un diccionario de Italiano-Español a mano, pero me imagino que dice quitar el viejo 68008. Tampoco lo he probado aún, pues tengo la Gold Card instalada en un QL MGE estándar y unidades de disco ED, y ya sabéis que esto ocupa el mismo espacio que cuando trabajabamos con un QL sin expandir. Eso sí, si se me cruzan los cables y decido comprar una nueva tarjeta sucesora de la Gold Card (daría mi Trump Card como entrada, os lo aseguro) o una tarjeta gráfica, habría que probar primero si es verdad eso de que la Gold Card funciona bien dentro de la caja Spem quitando el MC68008.

Salvador MERINO, Fuengirola, 1992 02 19

SE BUSCA

Se busca los siguientes programas de arquitectura para QL, vivos o muertos: ARTED, CIMED, COLUMNS TO BS 449, DESIGN TO BS 5628, EMPARED, GENED, PORED, SINGLE SPAW BEANS, SPACED, VICED, DEPURED, ECU, RAMED, RIDA, VIEWCONT... algunos

de ellos al parecer de las firmas TECORSA, CIBEL STRUC...
Además, se busca desesperadamente todo tipo de programas de gráficos, preferiblemente de tipo técnico, aunque también artístico.
Por último se busca, en este caso mejor vivo que muerto, alguien capaz de sacar una copia de pantalla decente con una BMC BX1000.
Mis datos, en el fichero suscriptores_txt.
Pablo POL, Madrid, 1992 02

CAZALETRAS

Presentación

En este juego se utiliza el ordenador únicamente como soporte físico del juego; él no "juega", sino que se dedica a comprobar que las jugadas sean lícitas, contar el tiempo y cosas por el estilo.
La idea es buscar palabras lo más complicadas posible, para que puntúen más, pero teniendo en cuenta que quien no pueda hacer palabra cuando llegue su turno, tendrá una penalización.

Ficheros

El juego se presenta con 5 ficheros, que se explican a continuación:
Cazaletras_boot => Se encarga de presentar brevemente las características del juego y lanzar el fichero ejecutable. Puede no usarse en absoluto, ya que el juego se puede lanzar de cualquiera de estas dos formas:
LRUN FLP1_cazaletras_bas
EXEC_W FLP1_cazaletras_exe
La pregunta que se presenta, "¿Quieres jugar?" se puede contestar afirmativamente con S, s, ESPACIO o ENTER. Negativamente con N, n o ESC
Cazaletras_bas => El programa fuente. Aunque no está estructurado en su sentido "puro", sí lo está en un sentido "impuro", es decir: el programa no contiene "GO TO", "GO SUB" ni "RESTORE xxx", con lo que se pueden eliminar todos los números de línea sin problema. Ahora bien, el programa utiliza varias variables globales y hay que tocarlo con un poco de cuidado.
Cazaletras_exe => Es el programa cazaletras_bas compilado con Turbo V2.04. He utilizado los siguentes parámetros de compilación:
Formato del fuente: Estructurado
Tamaño del objeto: <64K
Optimización: REMs
Números de línea: Omitidos
Número de ventanas SB a copiar: 3
Espacio para datos: 5K
Cazaletras_dat => Es el fichero que contiene los récords. Tiene que estar presente en FLP1_ mientras el programa corre, ya que si en algún momento se mejora el récord, el programa abre este fichero, para actualizarlo. El resto del tiempo permanece cerrado.
El formato del fichero es:
Repetido: (autor$\puntos\palabra$),
de modo que se puede modificar con un editor de textos ASCII, aunque hay que tener la precaución de dejar siempre al menos un bloque del tipo indicado.
El nombre de este fichero se puede modificar dentro del programa:
En cazaletras_bas, alterando en la línea 190 el nombre y en la 160 la longitud reservada.
En cazaletras_exe, editando el nombre con un editor.

Instrucciones

El programa pretende ser "auto-explicado", de modo que sólo expondré aquellos detalles que no aparecen en las pantallas.
La unidad utilizada por el programa se puede cambiar con facilidad, ya que se asigna en la línea 190 a una variable.
El juego se puede usar tanto en modo 8 como en modo 4, aunque es más rico en colorines en modo 8, como es lógico.
El juego tiene tres pantallas:
Primera. Sirve para captar los datos del juego. Las características del juego varían enormemente si se cambian los parámetros "Tiempo" y "Penalización". Os invito a probar distintas configuraciones. Si queréis fijar estos datos de otra manera, podéis cambiar las variables tope% y fallo% en la línea 390.
Segunda. Es la principal, en ella se desarrolla la competición. Observaciones:

-Se sale con ESC prácticamente en cualquier momento.

-Se puede borrar la última letra introducida con IZQUIERDA y también con CTRL/IZQUIERDA

-La palabra que se introduce con el tablero de letras lleno, es decir: la "primera" palabra de cada cuadro, sólo puntúa la mitad, porque si no tendría mucha ventaja el jugador al que le correspondiera.

-Las letras hay que introducirlas en minúsculas.

-La "K" y la "W" no aparecen nunca, de modo que no se pueden ni escribir, el programa no las admite.

-La "Ll" y la "Ch" no se han tenido en cuenta como letras dobles, sino que siempre cuentan como dos letras independientes.

-Las vocales acentuadas no están contempladas.

-La decisión de aceptar o no cada palabra se deja al criterio de los jugadores. Por tanto, el programa sólo comprueba si la palabra se puede construir con las letras que quedan en el cuadro.

Tercera. La opción "Continuar" "recuerda" el tiempo que le queda al jugador, pero no la palabra que había introducido.

Breve historia

Este epígrafe es un rollo, así que NO lo leáis.

El juego se me ocurrió mirando los distintos juegos de "letritas" disponibles en las jugueterías. Como una de las cosas que más me gustan del ordenador es que en disquetito se pueden "meter" un montón de fichas, dados, tableros, relojes... sin que ocupen mucho, me puse a escribir un juego que pudiera ser también construido físicamente. A las 48 horas tuve mi primera versión disponible. La frecuencia de aparición de las letras y su valor están relacionados con la frecuencia con que aparecen en castellano, aunque supongo que los valores que yo he asignado no serán absolutamente exactos.

Cuando compré Turbo, hice la segunda versión para que al compilarlo quedara bien. No tuve que cambiar gran cosa, y el programa seguía teniendo errores gordos: cada vez que se pulsaba una tecla se comía un segundo, y cuando se paraba el juego con ESC, no se podía volver al punto donde se estaba jugando.

Ahora que deseo donar el programa a QLíper, he escrito la tercera versión, que es la que tenéis. Entre otras cosas, he conseguido quitar los "GO" y que todas las líneas de código tengan menos de 80 caractéres, pero no muchas menos.

Derechos de autor

Ninguno en absoluto. Este programa es de dominio público y podéis hacer con él lo que os dé la gana. Eso sí, me gustaría que se reconociera, para bien o para mal, mi autoría:

Pedro REINA, Aguila 14 1º 2, E-28005 MADRID, España

Pedro REINA, Madrid, 1991 09 03

LOS MICRODRIVES Y SUS RUTINAS

Para acceder a los datos almacenados en los Microdrives, nos valemos del sistema de redireccionamiento de ES, así como de las rutinas especiales del QDOS, cuando necesitamos un mayor control, tanto en la lectura como en la escritura de la información en la cinta.

A continuación voy a tratar de exponer, de una manera detallada, y complemetando en lo posible el contenido del texto de A. DICKENS, todos los pasos a realizar para leer, escribir o verificar un sector de un microdrive.

PRIMERO.- SELECIONAR EL MICRODRIVE.

Para seleccionar un microdrive tenemos que proceder de la manera siguiente:

a).- Entrar en modo supervisor.

```
               moveq #$00,d
               trap #00
               movem.l a1/a6,-(a7)    * Salvamos contenido registros *
```

b).- Poner en a6 la base de las variables del Sistema.

```
               moveq #00,d0
               trap $01
               * Con este trap MT_INF ponemos en a0 el apuntador a la base
               de las variables del sistema, y a continuación lo llevamos
               a a6. *
               move.l a0,a6
```

c).- Esperar a que pare el microdrive.

```
label_1        move.b $00EE(a6),d0
```

```
                bne.s  label_1
                * Se utiliza la variable del Sistema SV_MDRUN, que
                se encuentra en $EE(a6). *
d).- Inhabilitar las interrupciones.
                ori.w  $700, sr
                * Instrucción solamente válida en modo supervisor. Así po-
                nemos a 1 los 3 bits ( 0, 1 y 2 ) de la máscara de interrup-
                ciones (byte alto, o byte del sistema ) del registro de es-
                tado SR. De esta manera colocamos la máscara a su nivel 7, es
                decir máxima prioridad, y quedan inhabilitadas las interrup-
                ciones, debido a que las peticiones de nivel igual o inferior
                a la máscara no son aceptadas. *
e).- Esperar hasta limpiar la transmisión serie.
label_2         subq.w  #01, $00A6(a0)
                blt.s   label_4
                move.w  #$208B, d0
label_3         dbra d0, label_3
                bra.s  label_2
label_4         clr.w  $00A6(a0)
                * Se utiliza la variable del sistema SV_TIMO. *
f).- Cargar en a3 el registro de control del microdrive, PC_MCTRL cuya
dirección es $28020.
                move.l  #$00018020, a3
g).- Poner modo microdrive.
                move.l #$FFFFFFE7, d0  * Desenmascara modo de transmisión *
                and.b  $00A0(a6), d0   * Desde modo compuesto *
                or.b  #$10, d0         * Pone modo microdrive *
                move.b d0, $FFE2(a3)   * En chip de periféricos *
                move.b d0, $00A0(a6)   * En variables del sistema
                * Este grupo de instrucciones necesita algunas aclaraciones.
                Tenemos que recordar cuál es el significado de cada uno de
                los bits del registro de control PC_TCTRL, localizado en la
                dirección $28002:
                bits 0-1 = velocidad de transmisón
                bits 3-4 = modo transmisión (10 = MDV seleccionado)
                bit 7   = salida directa
                Teniendo en cuenta esta información vemos que al hacer "and"
                con $FFFFFFE7 ( 11100111) ponemos a 00 los bits 3-4, modo de
                transmisión. Posteriormente al hacer "or" con $10 ( 00010000)
                ponemos a 10 los bits 3-4, modo microdrive.
                Recordar asimismo que $00A0(a6) corresponde a la variable
                del sistema SV_TMODE, y tener en cuenta que $FFE2(a3) es pre-
                cisamente $28002. Veamos: $28002 - $28020 = -$1E = FFE2 (con
                signo) = -30 (decimal). *
h).- Cargar en el Registro d1.w el número del microdrive que queremos se-
seleccionar.
                lea DAT_1, a0
                move.w (a0), d1
                * He utilizado  DAT_1 como dirección de constante definida,
                donde previamente se habrá colocado el número del MDV a se-
                leccionar. *
i).- Utilizar la siguiente rutina de selección.
                moveq #$03, d2         * Reloj alto. Bit sel. mdv puesto *
                subq.w #01, d1
                bsr.s  label_5
                bra.s  label_6
label_5         move.b d2, (a3)        * Reloj alto
                moveq #$3C, d0         * Tiempo de espera *
                ror.l d0, d0
                and.b #$01, d2         * Reloj bajo *
                move.b d2(a3)
                moveq #$3C, d0         * Tiempo de espera *
                ror.l d0, d0
                moveq  #$02, d2
                dbra, d1, label_5
                rts
                * Para mejor comprender este grupo de instrucciones, convie-
                ne también que recordemos el significado de cada uno de los
                bits del registro de control PC_MCTRL, localizado en la di-
                rección $28020:
```

```
                bit 0 = selecciona bit de mdv
                bit 1 = selecciona reloj de mdv
                bit 2 = ( 0 = leer, 1 = escribir )
                bit 3 = ( 1 = modo borrado )
                Así tenemos que al poner #$03 ( 00000011 ) en d2, ponemos
                a 1 el bit 0 y el bit 1, con lo cual seleccionamos reloj
                alto y mdv puesto. Al hacer "and" con #$01, lo que hacemos
                es poner a 0 el bit 1, dejando a 1 el bit 0. Al poner #$02
                ( 00000001 ) ponemos nuevamente a 1 el bit 1, reloj alto.
                Todo esto, claro está, refiriéndonos al registro de control
                &28020, contenido en a3.
                Finalmente con #$3C en d0, establecemos un período de espera
                de 150 ciclos. *
SEGUNDO.- BUSCAR UN SECTOR DEL MICRODRIVE.
Para buscar un sector determinado del microdrive, debemos realizar los si-
guientes pasos:
a).- Poner en d1.w el número de sector que buscamos.
label_6         lea DAT_2, a1
                move.w (a1), d1
                * Tambien aquí he utilizado  DAT_2 como  dirección de cons-
                tante definida,  donde previamente se habrá  puesto el número
                de sector requerido. *
b.- Cargar en a3 el Registro de control $28020. En este caso ya lo tenemos.
c).- Llamar a la rutina MD.SECTOR
                move.w #$03E8, d0       * Hacer 1000 intentos *
label_7         movem.w d0/d1, -(a7)
label_8         lea Buffer_1, a1        * Memoria intermedia *
                move.w $012A, a2        * Rutina MD.SECTR *
                jsr $4000(a2)
                bra.s  label_9          * Medio erróneo *
                bra.s  labl_8           * Cabecera inválida *
                movem.w (a7)+, d0/d1    * Leído correctamente *
                cmp.b d1, d7            * Comprueba sector *
                dbeq d0,  label_7
                bra.s  label_10
label_9         movem.w (a7)+, d0/d1
                bra.s  label_11
                * Llegado este punto es cuando estamos ya preparados para
                la utilización de las RUTINAS que nos proporciona el QDOS:
                para leer, para escribir o para verificar datos. Voy a con-
                tinuar con la rutina de LEER, como ejemplo. *
TERCERO.- LEER UN SECTOR DE MICRODRIVE.
Para la lectura de un sector de micrdrive utilizaremos la correspondiente
rutina.
label_10        lea Buffer_2, a1        * Memoria intermedia *
                move.w $0124, a2        * Rutina MD.READ *
                jrs $4000(a2)
                bra.s  label_11         * Si ha fallado la lectura *
                and.w #$00FF, d2        * Ha leído correctamente *
                move.b d2, (a1)+        * d2= Número de bloque *
                move.b d1, (a1)         * d1= Número de fichero *
                bra.s  label_11
CUARTO.- "DESELECCIONAR" UN MICRODRIVE.
Para "deseleccionar" un microdrive determinado, debemos llevar a cabo las
siguientes operaciones:
 a).- Poner en a6 la base de las variables del sistema.
label_11        moveq #$00, d0
                trap #$01               * Rutina MT.INF *
                move.l a0, a6
                move.l #$00018020, a3
                * Hemos recuperado  los valores correspondientes en los regis
                tros a6 y a3. *
b).- Utilizar la siguiente rutina de "deselección".
                moveq #$02, d2          * Reloj alto. Bit sel. mdv quitado *
                moveq #$07, d1
                bsr   label_5
                * Al hacer d2 = #$02 ( 00000010 ) ponemos a 0 el bit 0 y a 1
                el bit 1, con lo cual seleccionamos reloj alto y mdv quitado,
                del Registro de Control PC_MCTRL. *
c).-Quitar el modo microdrive.
```

```
                  move.l #$FFFFFFE7, d0  * Desenmascara modo transmisión *
                  and.b $00A0(a6), d0    * Desde modo compuesto *
                  move.b d0, $FFE2(a3)   * En chip de Periféricos *
                  move.b d0, $00A0(a6)   * En variables del sistema *
                  * Estas instrucciones ya han quedado explicadas anteriormen-
                  te. *
```

d). Habilitar las interrupciones y poner en modo usuario.

```
                  movem.l  (a7)+, a1/a6  * Recupera contenido registros *
                  move  #$0000, sr
                  * Instrucción como ya he comentado, solamente válida en modo
                  supervisor, que tiene por objeto poner a 0 los 3 bits de la
                  máscara de interrupciones, es decir a su nivel de mínima
                  prioridad, habilitando así las interrupciones. Tambien
                  ponemos a 0  el bit 7 del SR, con lo cual pasamos a
                  modo usuario. *
```

PARA FINALIZAR.- DEFINICION DE CONSTANTES Y BUFERS.

Voy a definir como constantes el número de microdrive a seleccionar y el número de sector a leer. También se establecen los búfers para la lectura de la cabecera de sector y para la lectura del sector. Estas instrucciones pueden modificarse a gusto del consumidor y según las necesidades del programa a desarrollar.

```
DAT_1          DC.B  1               * Núm. de MDV ( ponemos el 1 ) *
DAT_2          DC.B  1               * Núm. de Sector ( ponemmos el 1 ) *
Buffer_1       DS.B  14              * Para la Cabecera de Sector *
Buffer_2       DS.B  512             * Para los datos del Sector *
               END
```

Confío en que con esta información haya quedado al menos un poquito más claro, la utilización de las rutinas que nos proporciona nuestro QDOS, para el acceso a la información de los microdrives.

Es lógico que se puede mejorar y modificar la estructura del conjunto de todas estas instrucciones, siempre y cuando se respete el orden de su funcionamiento.

He adoptado la secuencia que me ha parecido más ideal para una más fácil y cómoda lectura y comprensión.

Félix ALONSO, Burgos, 1992 01 23

C68 2.0

Ya tenemos en nuestra librería QLíper la última versión de este famoso compilador de C de dominio público. Las primeras versiones tenían muchos errores, pero la presente versión es muy operativa comparada con las anteriores y algunas versiones C comerciales para QL. Además es muy probable que en futuras versiones sea tan potente que muchos usuarios escriban sus nuevos programas con el compilador C68 en vez de en SuperBASIC (el mismo Tony TEBBY ha preparado una librería QRAM y QPTR de dominio público para el C68).

Actualmente la versión mínima del C68 consta de 3 discos (8 con ficheros fuentes).

No voy a exponer todas las diferencias y ventajas de una versión a otra, porque eso sería ponerme a traducir como un loco, y no poseo tiempo suficiente actualmente. No obstante, os voy a poner a continuación la versión oficial de cambios desde la v1.00 a la V2.00.

Salvador MERINO, Fuengirola, 1992 02 10

CHANGES FROM PREVIOUS ISSUES

This file is a log of the main changes between the various releases of the C68 Compilation system. It also documents known bugs and C68 related developments that are known to be in progress.

For each release the changes are grouped under 3 categories:

a) New features. These can be enhancements to existing programs or libraries, or the addition of new ones.

b) Distribution Problems. This means that files that should be on the distribution disks are missing or have the wrong name. As the production of the distribution disks has been automated errors of this type will hopefully become less frequent as releases progress.

c) Bug fixes. This are changes introduced because a program or library routine does not perform as it is meant to.

The known bugs list applies to the last release specified. New entries are added as bugs are found/reported and old ones removed as they are fixed. When bugs are fixed a relevant entry will be added to the release specific details.

The final section covers C68 related development activities taht are known to be under way. These facilities will be made available in future releases of C68 as they are completed. If you are developing any feature of note that could usefully be included in future releases of C68 then please register the fact with the distribution co-ordinator. A short summary and the name of the person doing the work will be added to this document so that others can be made aware of this activity to avoid duplicating effort.

NOTE. This document is structured so that changes will only affect the end of the document. If you have obtained an update to C68 and wish to print this document, then you only need to print from the relevant page.

Release 1.00

This was the first version to be made available. It went on limited distribution.

Release 1.01 (available 18/05/91)

This was the first version to go out in response to the Quanta article announcing the availability of C68 under QDOS.

Changes

1. C68 upgraded to Version 3 Patch level 3.

2. The Decus CPP updated to a more recent version

3. SLB program added to distribution.

Program Bugs Fixed

1. Problem with AS68 that caused "Illegal Directive" messages to be output by the Linker. It was caused by FB bytes in the output data not being output as two FB bytes as required by the SROFF format.

Release 1.02 (available from 25/05/91)

Changes

1. The "CHANGES_DOC" file added to the SOURCE 5 disk.

2. The C68 compiler enhanced to support the \x (hex) numeric escape sequence in character constants and strings.

3. The CC program would now delete output files if an intermediate compile phase fails.

Distribution Problems fixed

1. An extra byte (at address 83103) had creapt into the LIBC_A library. This caused the LD linker to fail with

Error 4 Illegal Section).

2. The RUNTIME 2 disk was missing the "OVERVIEW_DOC" file.

3. The SOURCE 1 disk had the file C68_cglbldec_h incorrectly called "C68_cglbldev_h".

Program Bugs fixed

1. The AS68 assembler would generate an extra FB byte if one occurred in an XDEF SROFF definition. This could cause unpredictable problems at the link stage.

2. The C68 program was handling the numeric escape sequence \377 incorrectly (-1 was being treated as a special case!)

Release 1.03 (available 09/06/91)

Changes

1. PR utility added to distribution

2. SLB has new ANALYSIS option added.

3. A major rework of the header files was done to try and tidy them up. In particular they now make extensive use of the ANSI_H and SYS_TYPES_H header files.

4. The CC program has had a number of enhancements made:
   - There is now an option to direct the final file from the CPP, C68 or AS68 files to a temporary directory.
   - If you do a compile and link in a single call to CC, then any _o files created are also treated as intermediate files and deleted on successful completion.

Distribution Problems Fixed

1. The RUNTIME 1 disk had the file INCLUDE_wini_h incorrectly called INCLUDE_win_h.

2. The RUNTIME 2 disk was missing the libqram_doc file.

Program Bugs fixed

1. Yet another bug in AS68 found which could cause erronwous $FB bytes to be generated. This was when they occurred within an XREF sequence.

2. The CC program was giving the wrong name to final output files if the filename contained an extra underscore.

3. The CC program was passing the wrong name to the LD linker if LD first pass to be called by CC.

Release 1.04 (available 20/06/91)

Changes

1. The CPP pre-processor was translating -dvalue to upper-case (i.e. as though it was -DVALUE). This has been changed to preserve case.

2. The new routines 'ut_err', 'ut_err0', 'ut_window', 'ut_con', 'ut_scr', 'ut_mint', 'ut_mtext', 'ut_link' and 'ut_unlnk' added to the LIBC_a library. These map onto the corresponding QDOS vector routines.

3. The CC program will now accept the -i and -d options in both lower and upper case. This gives better

compatibility with the LATTICE implementations.

4. All the Assembler (_s) routines in thes standard C library should now be compatible with AS68.

Distribution Problems Fixed

1. The UNARC_BAS program mentioned in the README_DOC file was missing.

Bugs fixed

1. The routines 'io_delete', 'io_format' and 'mt_dmode' in the LIBC_a library would corrupt registers that might be used by C68 as register variables. This could cause unpredictable results. As 'io_delete' is called as part of 'fopen' and 'open' this was quite likely to happen.

2. The AS68 program has revealed an (obscure) bug in C68. The source was changed to avoid it. The result had been an invalid SROFF output file.

3. It was possible to get an "Error 10" when linking if your program contained no unitialised global variables. The LD statistics would show a BSS section with a strange start address and a size of zero.

4. The 'fseek', 'lseek' and 'fs_pos' routines in LIBC_A were working incorrectly if you specified SEEK_END as the mode. They would leave you at your current position.

Release 1.05 (available 14/07/91)

Changes

1. The tutorial has had a contents section, and chapters 13 and 14 added (plus associated example files).

2. C68 has been updated to Usenet patch level 4. The source now contains code to (optionally) generate 64 bit doubles. This has not yet been activated as the associated support routines are not yet ready.

Distribution Problems Fixed

1. Chapters 9 and 10 of the tutorial were incomplete

Bugs fixed

1. C68 was generating incorrect floating point constants. The C68 program was recompiled from scratch (including the support libraries) usijg first PDQC and the newly generated C68 and the problem disappeared. It must have been caused earlier by one of the library problems that have since been fixed!

2. A bug in the support routine for subtracting floating point numbers was corrected (it added them!).

3. The 'getc' macro was defined in a way that would cause end-of-file to be incorrectly reported if a character has a value of hex FF.

4. The versions of LIBMALLOC_A and LIBDEBUG_A needed some incorrect external references to be eliminated.

Release 2.00 (available 4th January 1992)

Changes

1. Memory allocation method upgraded to use QDOS common heap. Slave job now no longer used. All relevant library routines changed for new versions. All system programs recompiled and relinked with new libraries.

2. The Earl Chew stdio v2 package (ESTDIO) implemented in place of the previous stdio package in LIBC_A and LIBM_A. ESTDIO is a more complete implementation, and also conforms to the POSIX standard. This also fixes the problems with printing floating point numbers.

3. If a program is using a console for stdout then by default it is now cleared on starting the program, and at the end of the program a message is displayed and the program waits for a keypress before exiting (this does not occur if stdout is redirected, or inherited from an owning job).

4. Library documentation extended by adding summary lists to start of most documents. Other documentation rewritten and/or extended.

5. Tutorial Documentation improved. Added a guide to getting started with the tutorial.

6. A number of new routines added to libc_a and documentation updated accordingly:
   utime memincr stackmargin
   stackcheck memmax
   stackreport memqdos

7. CPP amended to ignore CR characters in input stream. A new -A option used so that it can be used in conjunction with assembler files.

8. The varargs_h header file added for Unix compatible programs that use this instead of the equivalent (slightly different) stdarg_h file introduced by ANSI C.

9. The defintion of the _PROTOTYPE macro in ansi.h changed to be more logical. All system include files changed to reflect this.

10. MAKE program upgraded to version 2. Documentation also amended so that examples reflect typical C68 working. Number of default rules built into MAKE now greatly expanded, and details given in documentation.

11. New programs UNPROTO, TSORT, UUENCODE and UUDECODE added to the standard distribution.

12. The CC front-end upgraded to support the optional UNPROTO pass, and to allow QMAC to be used as an alternative assembler to AS68. The -g changed to mean an implied new PROG_USE directory for all phases. The need for the -h option removed although it will still be accepted for compatibility reasons.

13. SLB librarian considerably enhanced:
    a) New -L option added to aid in producing optimum ordering of modules in libraries.
    b) New -W option added to produce a cross-reference by externals reference name.
    c) New -Y option added to produce cross-reference by file/module name.

14. Enhancements made to C68 program:
    a) Casts now taken into account when considering generating warnings for pointer type assignments.
    b) The identifer is now named when flagged as unknown.
    c) The default for the -warn parameter changed to ON as

a result of a) and b).

15. The routines fs_headr(), fs_heads(), io_fstrg(), io_fstrg(), io_sstrg() and sd_extop() in the LIBQDOS_A library have had the parameter order change. They timeout parameter is now the second parameter. THis makes them consistent with all other library routines which operate on a QDOS channel and have timeout as one of the parameters.

16. Additional string routines defined by ANSI (such as strcoll()) added to LIBC_A.

17. A number of new documentation files added:
    a) STARTING_DOC for guidance to new users of C68 to help[them get started quickly.
    b) SROFF_DOC containing details of the SROFF file format.
    c) TECHREF_DOC as a Technical Reference Guide for those who need details of how certain C68 features have been implemented.
    d) CMDINDEX_DOC as a quick index to the additional utilities provided with C68, each of which have their own DOC file.

18. An editor is now included in the standard distribution in the form of the QED Public Domain editor. This means that there is now no requirement for a separate editor to be able to use C68 (although of course any editor will do).

Distribution Problems/Changes

1. The RUNTIME set now consists of three disks:
    a) System Disk
    b) Utilities Disk
    c) Documentation Disk

2. The UTILITY_DOC file has been removed and replaced by individual files for each command.

3. TUTORIAL moved to its own disk, and no longer automatically included in standard distribution.

4. QRAM_H and LIBQRAM_DOC files removed from distribution. A new disk (labelled C68 QPTR library) will be produced that contains LIBQPTR_A, LIBQPTR_DOC and some examples of writing C68 programs to use the Pointer Environment. This has been contributed by Tony Tebby.

Bugs fixed

1. The LIBC_a routines to do with getting time were reporting a value 6 hours out. This was corrected by recompiling them with the current version of C68.

2. The ARC source as needed a few minor changes to compile with C68 (it was last compiled with PDQC). It would also ignore executable files if wild cards were used.

3. The stdarg_h file corrected. It did not handle parameter widening correctly.

4. A number of fixes made to the C68 program.
    a) An error in the support routine for floating point subtraction (it was doing an addition!) fixed.
    b) An error in allocating arrays larger than 32K fixed.

5. A number of system include files modified to bring them closer to the ANSI standard. Should not affect users.

6. The strchr() and strrchr() routines in LIBC_A were

incorrect (they were acting as if int was 16 bit).

7. A number of minor bugs in the STDIO routinese seem to have disappeared with the introduction of the ESTDIO package.

KNOWN BUGS

The following is a list of bugs/problems known to be present in the current release of C68.

1. The documentation for the Standard C library (LIBC_A) refers to a number of routines that are not present in this distribution . The routines affected are:

   stc... family of string routines
   stp... family of string routines

2. The documentation for the Maths library (LIBM_A) refers to a number of routines that are not (yet) present in the C68 distribution.

3. The CPP pre-processor does not always expand the __LINE__ macro correctly. This is most visible in the assert() error messages as the line number is always reported as 1.

PROBLEMS REPORTED

The following are problems that have been reported as being present, but which have not yet been investigated any further.

1. the 'read_dir' library routine is treating file type 2 as if it were a directory!.

2. The ARC utility accepts a maximum of 13 characters in a filename.

Developments being worked for future releases

The following is a list of various enhancements that are knwon to be in hand that will be incorporated in future releases of C68 as they get completed.

1. The enhancement of C68 to support 64 bit floating point to IEEE standard instead of the current 32 bit version. The work on the compiler has now been done, and the relevant support routines are being worked on. The 32 bit format may also be changed from the current "Fast Motorola Floating Point" format to the IEEE 32 bit format. (Christopher van Wullen).

2. The enhancement of C68 to support ANSI style function prototypes and parameter checking. (Keith & Dave Walker).

3. The enhancement of C68 to improve the error reporting system. (Keith and Dave Walkker).

4. The development of a semaphore library (Jeremy Allison)

5. A port of the ELVIS editor. Elvis is a Public Domain VI clone for those who know Unix editors. (Dave Walker ?).

6. A port of the RCS v5.5 source control system. (Dave Walker).

7. A port of an AWK and a YACC program (Ian R. Bruntlett)

8. Changes to allow the C68 system to produce the assembler output in GST compatible format as well as the current AS68 one. This will require the version of the GST Assembler that will be distributed by Quanta. An enhanced version will be produced for use with C68 that support for names longer than 8 characerts (plus other miscellaneous changes). The CC program will be enhanced to allow this alternative assembler to be used.
(Dave Walker)

9. Support for Run-time Dynamic Link Libraries. This will allow a set of library routines to be loaded at BOOT time and shared by all C68 programs.
(Lester Wareham and Dave Walker)

10. A library (LIBQPTR_A) to support the QRAM Pointer Environment. Also includes a short tutorial giving examples of its use.
(Tony Tebby).

11. A library (LIBTHING_A) to provide the support for the THING system.
(Eric Slagter and Lester Wareham)

12. A simple graphical front-end for running the C68 development system. Rather similar to the style adopted by the TURBO SuperBasic compiler.
(Richard Johnson)

ARC

Hace ya muchos años, los primeros usuarios de módems que trabajaban por aquel entonces bajo CP/M, inventaron un compresor/descompresor de paquetes llamado ARC. Esto ahorraba mucho dinero en la factura del teléfono, pues normalmente llega a reducir el tamaño de los ficheros en un radio de un 33% a un 50%. Tal ha sido su éxito y aceptación que existen hoy en día versiones compatibles ARC para casi todos los ordenadores y sistemas operativos.

No estoy muy seguro, pero me atrevería a decir que ARC fue el primero, y luego salieron otras variantes conocidas como ZIP, LZH, ZOO, ARJ,.... (todas disponibles para QDOS en dominio público, pero no presentes en QLíper aún).

Existen varias versiones ARC para QDOS, pero os he colocado con este artículo la versión que viene con el compilador C68, pues me parece muy completa. Para utilizarla necesitáis el TK2. Ejemplos de uso:

EX 'arc';'aqram1_ ram2_destino_arc ram1_*'

Empaqueta en el fichero ARC destino todos los ficheros contenidos en el dispositivo ram_1.

EX 'arc';'x ram2_destino_ARC'

Desempaqueta todos los ficheros contenidos en destino_ARC y los deja en el directorio corriente (DATA_USE).

EX 'arc';'x ram2_destino_ARC mi_programa'

Solamente desempaqueta mi_programa.

Comprendo que esto os va a recordar el intérprete de comandos del CP/M, MS-DOS o UNIX, y resulta un poco difícil para un usuario QDOS que no está acostumbrado a eso, pero como veis el SuperBASIC no se queda corto emulando esa forma de trabajar. El programita UNarc_bas, os ayudará a desempaquetar si tenéis algún problema.

Uno de los usos que yo le estoy dando al ARC es archivar en menos espacio mis copias de seguridad, pues en un disco DD caben 2 discos DD, en uno HD 4 discos DD, y en uno ED 9 discos DD. Como veis, esto puede llegar a ahorrar mucho sitio, pues a mi ya me resultaba bastante incómodo tener tanto disco.

En 1986 con la caída de Sinclair, me decían que tendría que tirar el QL por falta de programas. Hoy eso no lo tengo tan claro, y más bien tengo problemas de almacenaje de discos con software QDOS.

Ya solamente me queda decir que el tiempo de empaquetar/desempaquetar ficheros ARC con una GOLD CARD es inferior al tiempo empleado por una TRUMP CARD

en copiar ficheros de disco a disco con WCOPY.

Salvador MERINO, Fuegirola, 1992 02 10

[Se incluye el programa en el disco, que consta de los ficheros: arc_doc, arc y unarc_bas]

IMPRESOR PARA PROFESIONAL PUBLISHER

En nuestra librería QLíper existe un programa en el disco I002 que permite imprimir páginas 960x(1-800) en impresoras compatibles EPSON de 24 agujas. El autor lo escribió, porque el driver de 8 agujas que trae el Profesional Publisher no imprime bien en impresoras de 24 agujas debido a una diferencia de escala (1/72 en 8 agujas y 1/60 en 24 agujas). Ese mismo motivo fue el que me hizo a mí comprar el programa GRAFIX v3.0 cuando cambié mi BMC por mi EPSON LQ-550.

La nueva versión, que envío junto a este artículo, posee una tercera opción escrita por mí. Lo único que he añadido es una nueva versión del fichero PIN_BIN (PIN8_BIN) para 8 agujas en vez de 24, y una versión SuperBASIC, Publisher_esc_l_bas, para impresoras 8/9 agujas que soporten el modo gráfico ESC L compatible EPSON (p.e.: La BMC).

Hasta hoy nuestra revista, QLíper, se ha hecho usando Quill con muy buenos resultados, pero si como pienso, la mayoría de nosotros tenemos una impresora de 8/9/24 agujas y un QL con un mínimo de 384 Kbytes de RAM, a partir de ahora podría diseñarse la revista con el Profesional Publisher y usar el programa impresor en PD para imprimirla cada uno en casa. Naturalmente, las páginas deberían ir comprimidas en formato ARC, pues en caso contrario, iban a caber pocas páginas en un disco 3.5" 720K. Con todo esto, nuestra revista ganaría muchísimo en calidad de presentación.

Salvador MERINO, Fuengirola, 1992 02 11

[Se incluye en el disco este programa, que consta de los siguientes ficheros: PUBLISHER_BAS, PIN_BIN, Stampa_bin, boot_publisher, NOT_BIN, Publisher2_bas, 24PIN_EXT, pin8_asm, publisher_esc_l_bas, pin8_bin, y ejemplo_page]

¿COMO INSTALAR UN MODEM HAYES EN EL QL?

Hoy en día la mayoría de los MODEMs externos compatibles HAYES son inteligentes, y han bajado mucho de precio (p.e.: mi módem externo ECSA CJ2.4). Para el QL es muy recomendable adquirir uno inteligente, pues entre muchas cosas, puede responder él solito una llamada.

Afortunadamente el cable a utilizar para conectar el módem al QL es el mismo que el de la impresora serie, pero conectado en SER2.

Desafortunadamente el QL solamente dispone de 4 señales en el interface serie, no tiene DTR y DCD. En mi módem estas señales se pueden poner siempre en "ON" activando unos micro-interruptores (el 5 y el 6). También es posible hacerlo por software con los siguientes comandos HAYES:

AT &C0

Indica que la linea 109 (DCD) estará permanentemente activada.

AT &D0

Indica ignorar el estado de la línea 108 (DTR) y suponerla permanentemente activada.

Como veis, existen pocos problemas a la hora de conectar un QL a un módem compatible HAYES e inteligente via RS-232. Además, todo el software de comunicaciones lo tenéis esperando en nuestra librería QLíper.

Salvador MERINO, Fuengirola, 1992 01 21

GOLD CARD: PRIMERAS IMPRESIONES GENERALES

¡He recibido la Gold Card!

Mi equipo en este momento consta de dos QLs, uno equipado con Gold Card y Light-ning en ROM, y el segundo con Expanderam de 540 Kbytes y Toolkit II en ROM.

Al colocar la Gold card, la reducción de ta-ma-ño permite que la am-pli-a-ción
esté to-tal-mente dentro de la caja del QL lo que significa, aparte de la dis-po-ni
-bi-li-dad de espacio y mejora en el go-bier-no, una mejor protec-ción fren-te

a aco
-metidas exteriores.
Al usarla, el aumento de velocidad es evidente en todos los programas de Psion que he probado, así como en, The Painter, Desktop profesional publisher, Text87, Perfection, y algunos míos de uso corriente en Basic, directamente y turbocompilados.
No he probado el interface serie/paralelo; tampoco la Minerva (no la poseo), sobre la que aparecen noticias de algún tipo de incompatibilidad. Por ejemplo, que con Minerva + Lightning el ordenador se cuelga en programas Basic con la secuencia "OPEN#3,SER1", pero no lo hace si se suprime Lightning; también hay problemas de paginación con Archive (Ian Pizer, de Ginebra en Quanta v8 i10 página 29).
La primera impresión personal es que te han cambiado los pro-gra-mas; o que se han tranformado, que son otros.
Quill pasa a ser un tratamiento normal; yo uso Turbo Quill+, y re-al-mente pasa a ser un programa muy rápido para pequeños trabajos de unos pocos folios; no lo he probado con trabajos largos.
Donde he apreciado más utilidad es en el manejo de programas re--lativamente largos para tratamiento de ficheros con Archive (clasificación y cálculo); aunque no he utilizado cronómetro, el tiempo de tratamiento se re-duce aproximadamente a un tercio.
En Abacus hay aumento de rapidez, pero he encontrado dos problemas, por una parte no lee algunos archivos hechos con el equipo anterior, y por otra que se produce un cambio en la ventana informativa consistente en que el con-tenido de cada celdilla no aparece, como sería normal aparte del in-di-ca-ti-vo "CONTENTS
",
sino a continuación del contenido de "MEMORY". Esto no su-ce-de con la reducción de memoria a 128 K. No he apreciado error en los cál-cu-los.
En The Painter la velocidad aumenta tanto que lo hace inmanejable; usan-do los ajustes de usuario mejora algo, pero no es suficiente. Sin embargo el comando general SLUG lo resuelve. Con SLUG 15 la velocidad es, para mi modo de usarlo, correcta. No he ensayado bien el ratón, ya que en un momento el cursor apareció con mo-vi-mien-tos erráticos incontrolables, y no he comprobado a qué
se
deben, si a un fallo propio o a la nueva configuración.
Desktop Profesional Publisher funciona normalmente sin especiales mo-di-fi-
ca
-cio-nes. Es quizá donde menos se aprecia un aumento de velocidad. He observado algún problema para leer ficheros anteriores; no eran importantes, y los borré; los nuevos los lee sin problemas.
Los programas Basic siguen tardando en cargarse, pero no obstante la ejecución es más rápida tanto directamente como compilados.
El aumento de memoria ram me resulta especialmente útil con Text87 ya que al usarlo conjuntamente con una base de datos y una hoja electrónica, la dis
-ponibilidad adicional de memoria es muy importante. Con 640K en trabajos largos me veía obligado a fraccionarlos en capítulos,mientras que con la ampliación hay mucha holgura y se pueden tener simultáneamente otras utilidades como QPACKII sin problemas.
Globalmente el juicio es claramente favorable. No he apreciado problemas importantes en los programas que yo uso habitualmente, ha aumentado la capacidad de memoria, con la que antes tenía limitaciones, y además funcionan todos los programas más rápido. Para mí ha sido la solución a medida.

TEXT87 4 PLUS

Esta versión fue anunciada en el QL World de mayo de 1991.
En esas fechas remití a Software87 el disco original de la versión 3 pa-ra obtener la nueva versión mediante pago con tarjeta Visa, pero hasta la fe--cha no he recibido ni el programa, ni noticias, ni la devolución de mi en--vío.
Tampoco he visto ni críticas ni nuevos anuncios, ni en QL World, ni en Quanta, ni en QL-Contact France.
Quizá se trate de una información sonda para explorar la demanda potencial antes de ultimar la versión, o para restar comercio a otros programas como pudiera ser Perfection.

BIBLIOTECA QLíPER

Me resulta muy interesante la inclusión de una biblioteca.
Yo tengo bastantes discos de Quanta y de QL-Contact France, de las que soy socio, no obstante casi en plan de coleccionista, pues mis ocupaciones ac--tuales

me absorben entre 14 y 15 horas diarias. Por tanto están en caji-tas, tal como llegaron. Dado que me jubilo dentro de poco, espero que entonces puedan constituir un alivio al trauma que siempre provoca el dejar la vida activa de trabajo. ESTOY DISPUESTO A ESTUDIAR CUALQUIER FORMULA QUE SE CONSIDERE CONVENIENTE PARA EL GRUPO.

Supongo que otros miembros estarán en contacto o serán socios de alguna de las agrupaciones existentes en otros países, por lo que, aunque todos establecen claras limitaciones a la difusión, dadas las características especiales de nuestro grupo, sería interesante estudiar alguna fórmula de uso EN COMUN, ya sea directamente, o bien llegando a algún tipo de acuerdo como grupo.

Lo que sí hacen los franceses son compras en común a precios claramente interesantes. Yo he aprovechado las ofertas en algunas ocasiones y funciona correctamente. Es necesario anotarse previamente pues ellos hacen el pedido sobre compromiso firme.

En Quanta de noviembre de 1991 se hace referencia a SOFTWARE DE DOMINIO PUBLICO PARA QL con un coste de 50 peniques por copia. Dan dos entidades que lo copian; una de ellas C.G.H. sin dirección, pero la otra con la dirección siguiente:

QUBBESOFT P/D, 38 BRUNWIN ROAD, RAYNE, BRAINTREE, ESSSEX CM7 5BU
TEL 0376 47852

Yo no poseo ese software de dominio público; tampoco tengo la relación de programas y por tanto desconozco si tiene interés para nosotros.

PUEDE PENSARSE EN UN AUMENTO DE CUOTA si fuera necesario para cubrir suscripciones o gastos de pedidos aunque bien sé que esto no compensa el entusiasmo, la dedicación y el sacrificio personal de quienes tenéis la carga del trabajo de llevar adelante el club. YO TENGO FACILIDADES Y PUEDO COLABORAR, POR EJEMPLO HACIENDO FOTOCOPIAS DE INSTRUCCIONES, sin más cargo que el coste por página del papel y máquina (del orden de 1,90 ptas.) siem-pre que estemos hablando de la dimensión actual del grupo.

Saludos afectuosos.

Dasio CARBALLEIRA, Santiago de Compostela, 1991 11 01
Tel.: 981-565974 Fax: 981-563002/573723

¡EL ASCII HA MUERTO! UNICODIGO E ISO 10646 PELEAN POR LA SUCESION

ASCII, el código estadounidense para intercambio de información, es el código más difundido en el mundo. Pero en los últimos años se ha visto que el ASCII no sólo no es suficiente, sino que acarrea peligros para el comercio. Cuando un banco en Estados Unidos envía el código del carácter "$", éste llega a Gran Bretaña como "". ¡Los dólares estadounidenses se convierten en libras esterlinas!

El sucesor del código ASCII se llama Unicódigo [Unicode en inglés, Unikodo en esperanto], esto es lo que dice un consorcio de grandes firmas informáticas. Este código universal solucionará pronto los problemas del intercambio de información a nivel internacional. El nuevo código contiene 27000 caracteres, que representan todas las principales lenguas del mundo. Por supuesto, incluye el esperanto. Pero también ISO, la Organización Internacional de Normalización, ha preparado un nuevo código, denominado ISO 10646. ¿Cuál de ellos será el vencedor?

Un poco de historia.

En 1983 ISO empezó a trabajar para desarrollar un sucesor del ASCII. Este nuevo código habría de tener, según estaba planeado, todos los caracteres de todas la principales lenguas del mundo. ISO reconoció que los códigos más usados hasta entonces, el ASCII y el EBCDIC, no bastaban.

El ASCII fue aceptado por ANSI (Instituto Nacional Estadounidense de Normas) en 1977. ISO adoptó este código de 7 bits prácticamente sin ningún cambio y lo llamó ISO 646. Esta misma asociación después amplió el código en un bit, haciéndolo por tanto de 8 bits, lo cual le permitió contener hasta 256 caracteres (2^8=256). Este código contenía entonces las letras que necesitaban algunas lenguas europeas como el francés, el español y el alemán. Este código se conoce con el nombre de Latín 1. A éste le siguieron otros sistemas de códigos para lenguas descendientes del latín: Latín 2, Latín 3, Latín 4. Los países de Africa y Asia adaptaron este código según sus necesidades. Para representar los caracteres del japonés, chino y coreano, fueron evolucionando diversas normas.

También los esperantistas conocen los problemas de representar caracteres que no existen en el código ASCII. Para ello han usado programas como Lettrix, han adaptado programas existentes como WordPerfect, han obligado a los productores de logicial a incorporar en nuevas versiones las letras del esperanto (como pasó en el caso de Locosript 2), han escrito sus propios programas, como Tekstoro, o simplemente se han resignado. Una parte de toda esa historia se puede leer el las páginas de Komputila Rondo de los últimos cinco años.

Y también, los costes de adaptación de los equipos y de los programas se hicieron enormes para los fabricantes internacionales. La única posibilidad era crear un código nuevo.

Problemas durante el desarrollo.

Cuando ISO empezó a crear un nuevo código, se decidió que fuese de 16 bits (2 octectos para cada carácter), lo que permitiría representar 65536 caracteres (2^16=65536). Se perseguía un código que fuese compatible con todos los códigos existentes. Los primeros 128 caracteres estaban reservados para el ASCII, y los demás juegos de caracteres debían ir a continuación. Aparecieron dos problemas: los caracteres de control y la representación de los caracteres Han.

Los caracteres de control son usados por muchos programas de comunicaciones. Por eso se decidió conservarlos en el nuevo código, pero esto costó el 40% de los 65536 espacios, antes de asignar ningún otro carácter.

Los caracteres Han (de la dinastía china Han) se usan en chino, japonés y coreano. A menudo los caracteres son los mismos o muy similares en los tres idiomas. Por este motivo ISO decidió desarrollar un juego de caracteres Han unificado para todos, pero ni Japón ni Corea lo aceptaron. Entonces no quedaba más remedio que meter todos los caracteres en el nuevo código, con el resultado de que no había sitio para todos en los 65536 espacios disponibles.

ISO, después de largas discusiones, decidió ampliar el código a 32 bits (4 octetos por carácter). Pero entonces muchos pensaron que un código así ocuparía demasiado espacio. Se propuso asignar octetos de ordenación, que permitirían enviar sólo un octeto si los anteriores no cambiaban. La propuesta no fue aceptada porque se temió que los octectos de ordenación complicarían demasiado el código, ya que los caracteres no siempre se representarían con el mismo número de octetos. Los problemas frenaron cada vez más y más el desarrollo.

Unicódigo.

Ya que desde hacía 4 años los diversos intentos no habían logrado un código internacional, un grupo de empresas decidió crear un código propio que fuese sencillo pero también más consistente que el ISO 10646. El asunto empezó con Xerox y Apple, pero pronto se unieron Metaphor Computer Systems, Research Libraries Group, Microsoft, Sun, Adobe, Claris, Next, y Pacific Rim Connections. Este grupo de alto nivel deseaba que se crease rápidamente un código internacional.

Se creó el nombre Unicódigo. El nombre había de expresar que el código es único, universal e uniforme. Asimismo había de ser:

-COMPLETO. Ha de contener todos los caracteres necesarios para el tratamiento de textos, incluso los de las lenguas muertas.

-SIMPLE Y EFECTIVO. Cada carácter ha de estar siempre representado por 16 bits. No se permiten caracteres de control.

-INEQUIVOCO. Cada código de carácter representa exactamente un carácter.

-CONVERTIBLE. Un texto representado con el nuevo código ha de ser convertible a los antiguos códigos.

Estos objetivos guiaron el Unicódigo, el cual tiene espacio para 65536 caracteres. Los primeros espacios están reservados para el ASCII. Y sólo ahí hay caracteres de control, aunque no se usan en Unicódigo. La siguiente lista muestra cómo se distribuyen los espacios en Unicódigo:

0-8192	alfabetos
8193-12287	símbolos
12288-16383	alfabetos chino, japonés, coreano, y signos de puntuación
16384-59391	ideografías china, japonesa y coreanas
59392-65024	espacio a disposición del usuario
65025-65535	espacio para datos de conversión

Los primeros 8193 están reservados para los diversos alfabetos. Queda espacio para alfabetos de lenguas históricas (por ejemplo los jeroglíficos egipcios). 4094 espacios están ocupados por símbolos como signos de puntuación, símbolos técnicos e incluso decorativos, religiosos, o piezas de ajedrez. Los alfabetos chino, japonés y coreano y los signos de puntuación ocupan 4095 espacios. Los siguientes 43007 espacios están reservados para el juego de caracteres Han unificado y para usos futuros. Unicódigo permite al usuario crear hasta un máximo de 5632 caracteres propios. Por último, 510 espacios se ocupan con los datos necesarios para hacer conversiones a Unicódigo.

Código y aspecto.

Unicódigo distingue entre el código y el aspecto de un carácter. El código nunca cambia, pero el aspecto puede cambiar un poco, dependiendo del tipo de letra que se use. El aspecto básico es el mismo.

¿Unicódigo o ISO 10646?

A pesar del hecho de que ISO 10646 tiene más años que Unicódigo, la confianza en ISO 10646 en los países que colaboran con ISO no es evidentemente demasiado alta. En 1991, en la decisión acerca de la adopción de un código internacional, 10 países votaron en contra de ISO 10646 (entre ellos, Estados Unidos).

Con el éxito de Unicódigo, que en parte es el fracaso de ISO 10646, ahora sólo hay dos posibles códigos para ser los herederos del ASCII. El coonsorcio Unicódigo dice que está intentando imponer Unicódigo como norma internacional. Esto parece necesario, porque muchos Estados prefieren una norma reconocida por ISO a una norma de hecho creada por empresas privadas. Por eso ya, poco después del anuncio de la versión 1.0 de Unicódigo, el consorcio habla de Unicódigo 1.1, que tendría los cambios necesarios para la adaptación a ISO 10646. Pero después de esto, se ha dejado muy claro que ningún carácter podrá cambiar nunca su representación en el código.

Por tanto la respuesta a la pregunta "¿Unicódigo o ISO 10646?" será: "Unicódigo e ISO 10646".

Consecuencias.

Un nuevo código traerá consecuencias para toodos. Los fabricantes de ordenadores tienen que implementarlo en sus programas básicos. Las firmas de logicial tienen que adaptar sus programas a la nueva norma. Esto tal vez durará entre uno y dos años. ¿Y los usuarios? Ahora los usuarios tendrán que pagar todo esto. Si quieren (o tienen que) usar el nuevo código, por lo menos tienen que adaptar los aparatos que usan, ordenadores, tarjetas gráficas, impresoras, etc., o bien comprar unos nuevos. Y después es necesario comprar las versiones actualizadas de los programas que usan normalmente. Y además el nuevo código duplicará el tamaño de los textos ASCII. Pero debido a que cada vez menos documentos son simples textos ASCII, sino que incluyen gráficos, el espacio necesario para conservarlos será sólo un 25% mayor. Todo esto será sin duda un buen negocio para los fabricantes de equipos y programas. Esperemos que los costes para los usuarios no frenen la introducción del nuevo código.

Si quieres comprar un ordenador, compra sin ninguna duda uno que pueda manejar códigos de 16 bits, por tanto basado en un procesador 80386 o 80386SX. En otro caso no será posible adaptar el equipo a la nueva norma.

Las ventajas para el trabajo de los especialistas esperantistas son evidentes, porque en su trabajo los contactos internacionales son algo normal. Komputila Rondo estará pendiente del momento en el que los primeros productos que adopten esta norma se comercialicen. Ciertamente hay bastante material para nuevos e intersantes artículos.

Komputila Rondo 3/91
Traducido del esperanto por:
Marcos CRUZ, Madrid, 1992 03 02